Chapter 10
JavaScript: Arrays

Internet & World Wide Web
How to Program, 5/e

ftp://webdesign:webdesign@120.126.16.67:55555

OBJECTIVES

In this chapter you'll:

- Declare arrays, initialize arrays and refer to individual elements of arrays.
- Store lists and tables of values in arrays.
- Pass arrays to functions.
- Search and sort arrays.
- Declare and manipulate multidimensional arrays.

10.1 Introduction

- Arrays 陣列
  - Data structures consisting of related data items
    - 註：儲存相關資料的結構
- JavaScript arrays
  - Can “dynamically” (動態) change size (變更大小) after they are created
- An array is a group of memory locations
  - All have the same name (相同變數名稱) and normally are of the same type (相同型態)
    - 註：陣列是一群記憶體位置，有相同變數名稱、相同型態
    - 註：陣列內個別的記憶體位置，稱為”element (元素)”
    - 註：當要存取其中的element時，使用陣列名稱[編號]，其中中括號是描述第幾個位置的element。

10.2 Arrays (Cont.)

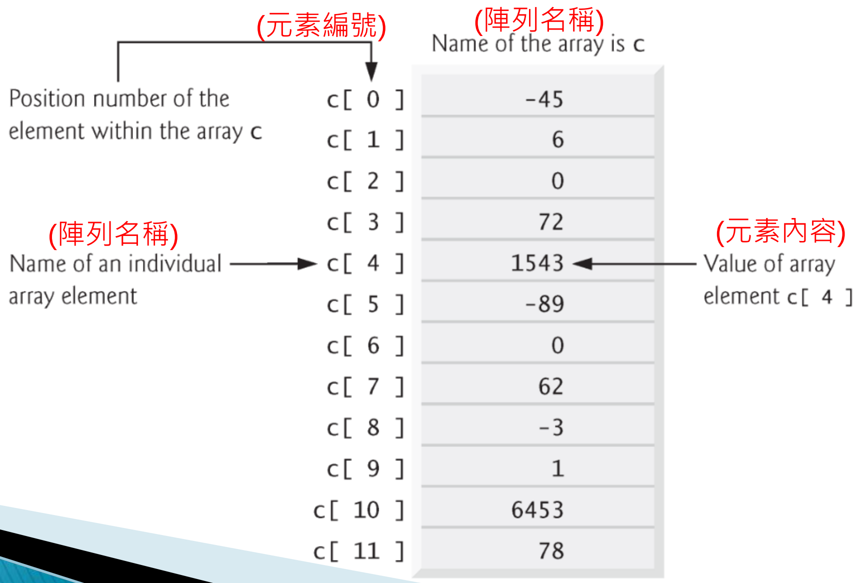
arrayname[元素編號] //陣列

- The first element in every array is the zeroth element.
  - 註：第一個element是編號0, ex: A[0]

- The *i*–th element of array c is referred to as c[i-1].
  - 註：第 i個元素的編號為i-1.

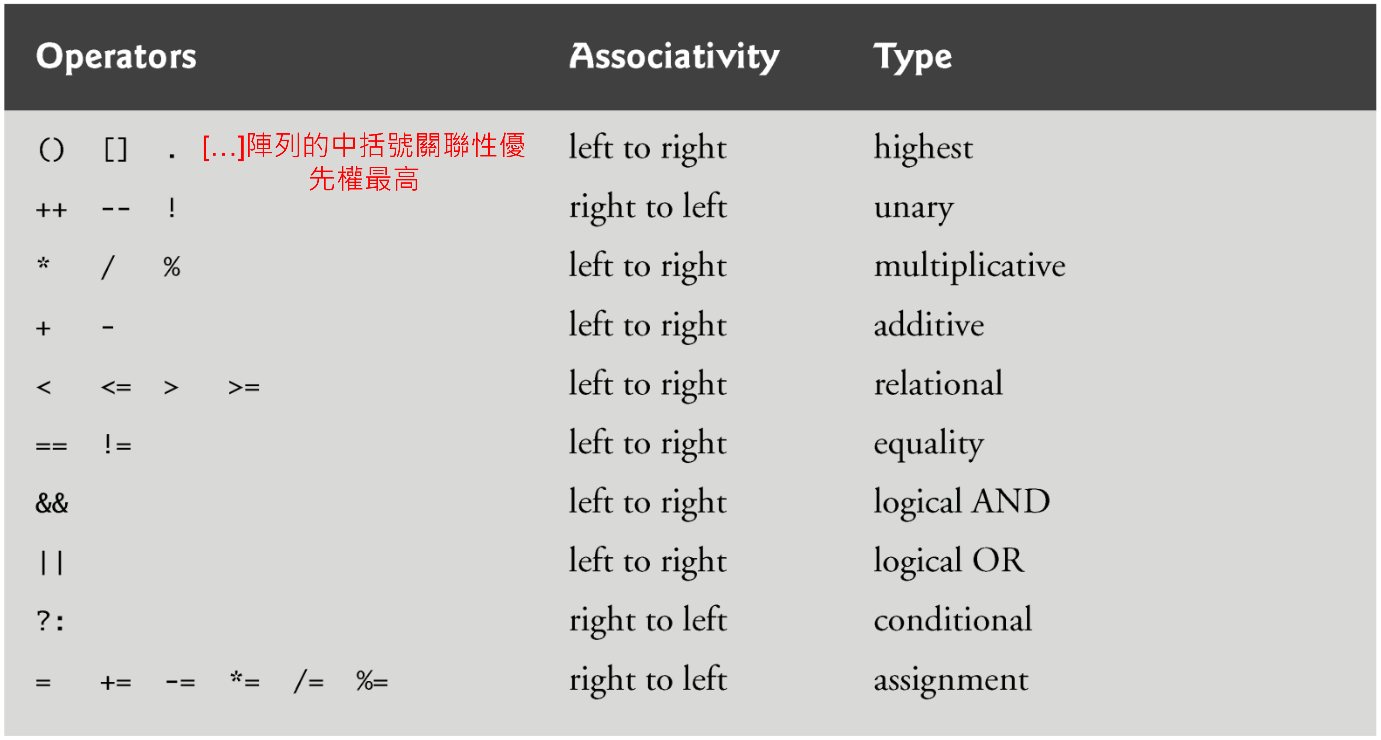
arrayname.length 陣列長度

- In JavaScript, every array knows its own length (長度可知), which it stores in its length attribute (長度屬性) and can be found


(元素編號)
(陣列名稱)
Name of the array is c
Position number of the element within the array c
(陣列名稱)
Name of an individual array element
(元素內容)
Value of array element c[4]

c[0]	-45
c[1]	6
c[2]	0
c[3]	72
c[4]	1543
c[5]	-89
c[6]	0
c[7]	62
c[8]	-3
c[9]	1
c[10]	6453
c[11]	78

10.2 Arrays (Cont.)



Operators	Associativity	Type
`()` `[]` `.` [...]陣列的中括號關聯性優先權最高	left to right	highest
`++` `--` `!`	right to left	unary
`*` `/` `%`	left to right	multiplicative
`+` `-`	left to right	additive
`<` `<=` `>` `>=`	left to right	relational
`==` `!=`	left to right	equality
`&&`	left to right	logical AND
`||`	left to right	logical OR
`?:`	right to left	conditional
`=` `+=` `-=` `*=` `/=` `%=`	right to left	assignment

Fig. 10.2 | Precedence and associativity of the operators discussed so far.

10.3 Declaring and Allocating Arrays

- JavaScript arrays are **Array objects (陣列物件)**.

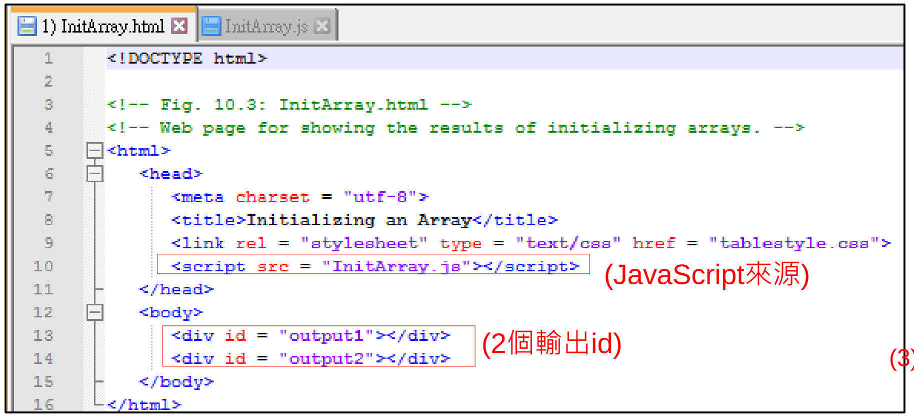
var 陣列名稱 = new Array(元素個數); //陣列宣告

- You use the new operator (新建) to create an array and to specify the number of elements (元素個數) in an array.
- The new operator creates an object (創建物件) as the script executes by obtaining enough memory (預留記憶體) to store an object of the type specified to the right of new.
  - 註：new是用來創建一個物件，使物件獲得充份的記憶體。

- JavaScript reallocates (重新配置) an Array when a value is assigned to an element that is outside the bounds of the original Array
  - 註：若寫入的元素編號超出原本宣告長度，JavaScript會直接重新分配置此陣列元素數

10.3 Declaring and Allocating Arrays

- Example:
  - 指定與未指定陣列長度

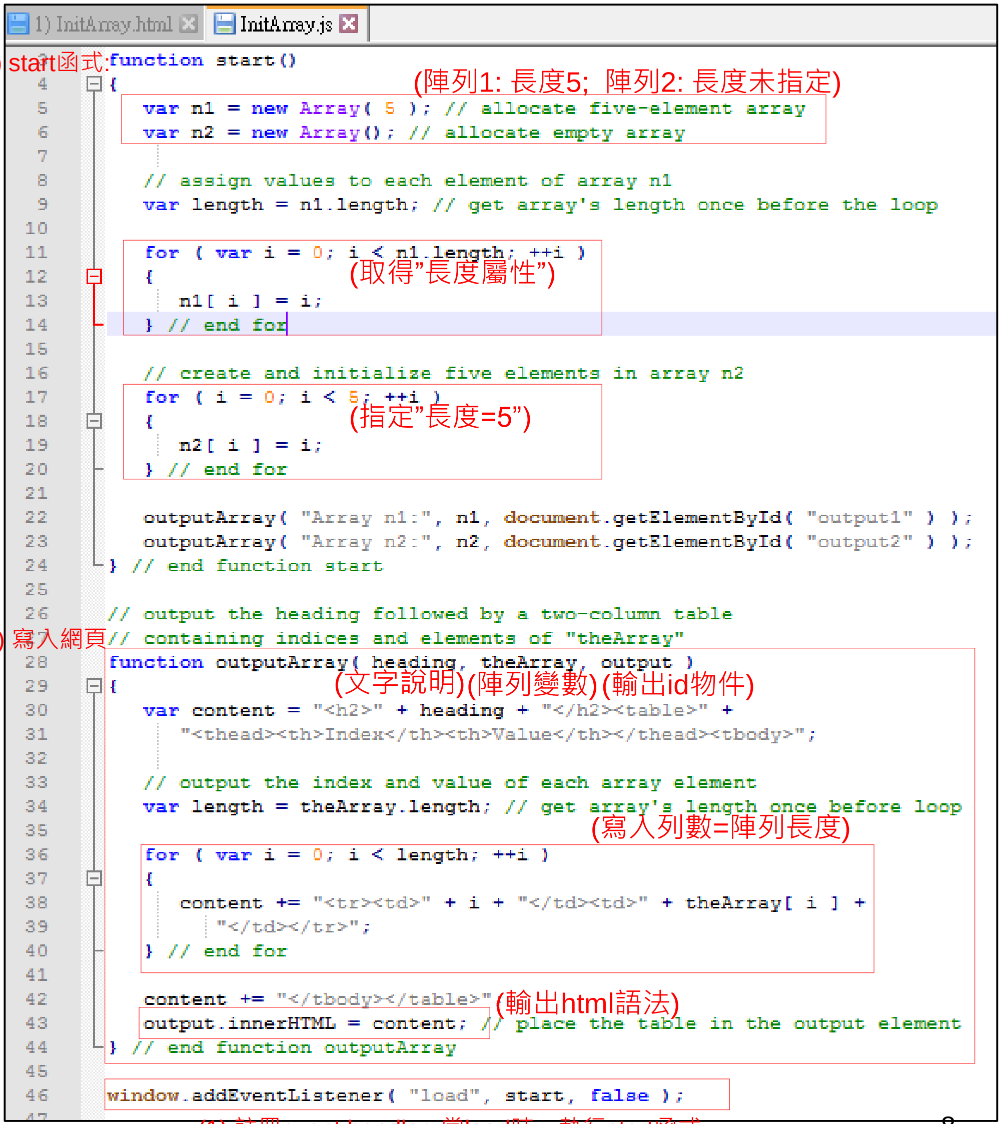


1) InitArray.html | InitArray.js

```html
<!DOCTYPE html>

<!-- Fig. 10.3: InitArray.html -->
<!-- Web page for showing the results of initializing arrays. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>Initializing an Array</title>
      <link rel = "stylesheet" type = "text/css" href = "tablestyle.css">
      <script src = "InitArray.js"></script>
   </head>
   <body>
      <div id = "output1"></div>
      <div id = "output2"></div>
   </body>
</html>
```

(JavaScript來源)

(2個輸出id)



1) InitArray.html | InitArray.js

```javascript
function start()
{
   var n1 = new Array( 5 ); // allocate five-element array
   var n2 = new Array(); // allocate empty array

   // assign values to each element of array n1
   var length = n1.length; // get array's length once before the loop

   for ( var i = 0; i < n1.length; ++i )
   {
      n1[ i ] = i;
   } // end for

   // create and initialize five elements in array n2
   for ( i = 0; i < 5; ++i )
   {
      n2[ i ] = i;
   } // end for

   outputArray( "Array n1:", n1, document.getElementById( "output1" ) );
   outputArray( "Array n2:", n2, document.getElementById( "output2" ) );
} // end function start

// output the heading followed by a two-column table
// containing indices and elements of "theArray"
function outputArray( heading, theArray, output )
{
   var content = "<h2>" + heading + "</h2><table>" +
      "<thead><th>Index</th><th>Value</th></thead><tbody>";

   // output the index and value of each array element
   var length = theArray.length; // get array's length once before loop

   for ( var i = 0; i < length; ++i )
   {
      content += "<tr><td>" + i + "</td><td>" + theArray[ i ] +
         "</td></tr>";
   } // end for

   content += "</tbody></table>";
   output.innerHTML = content; // place the table in the output element
} // end function outputArray

window.addEventListener( "load", start, false );
```

(2) start函式:

(陣列1: 長度5; 陣列2: 長度未指定)

(取得"長度屬性")

(指定"長度=5")

(3) 寫入網頁

(文字說明)(陣列變數) (輸出id物件)

(寫入列數=陣列長度)

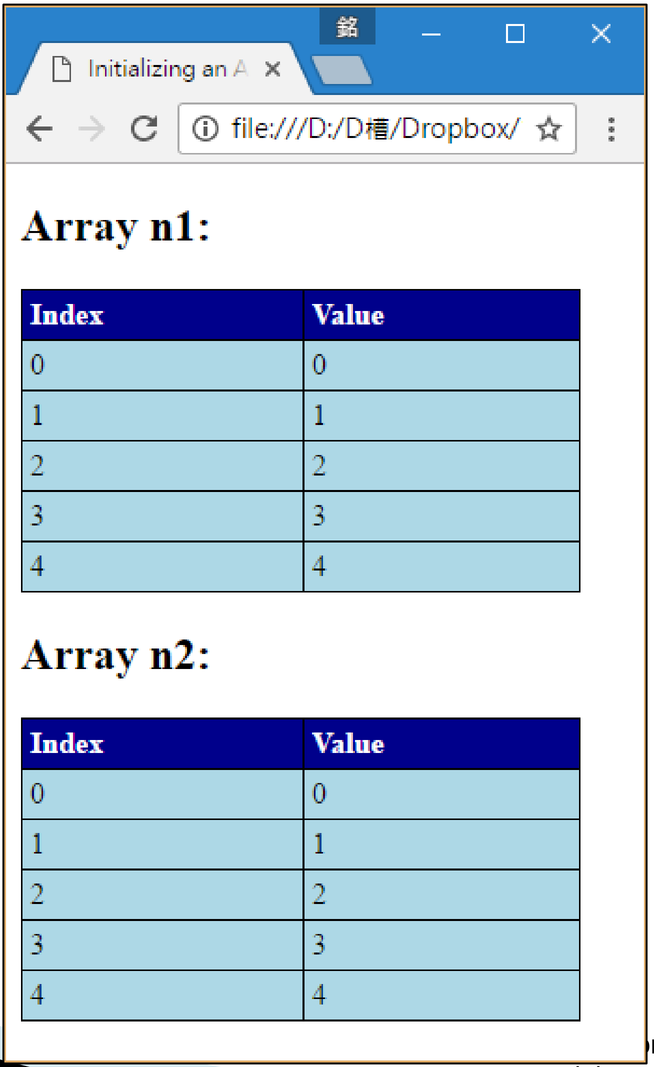
(輸出html語法)

(1) 註冊event handler: 當load時，執行start函式

10.3 Declaring and Allocating Arrays

- Output:

(輸出)



Array n1:

Index	Value
0	0
1	1
2	2
3	3
4	4

Array n2:

Index	Value
0	0
1	1
2	2
3	3
4	4

(陣列1: 長度5)

(陣列2: 長度也為5)

10.4 Examples Using Arrays (Cont.)

陣列宣告&初始化 (範列一)
var 陣列1 = [2, 4, 6, 8];
var 陣列2 = [2, , , 8];

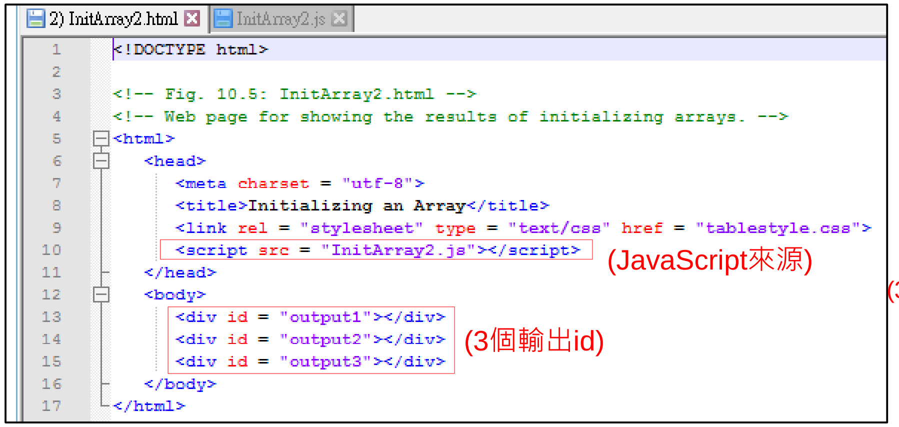
- Arrays can be created using a comma-separated **initializer list (初始序列)** enclosed in square brackets ([…])
  - 註：中括號內放入初始元素，利用”逗號”分開
  - The array’s size (陣列長度) is determined by the number of values in the initializer list
  - 註：陣列長度取決於幾筆數值放入初始序列。

陣列宣告&初始化 (範列二)
var 陣列名稱 = new Array("…", “… ","…", …);

- The initial values of an array can be specified as arguments in the parentheses (小括弧) following `new Array`
  - The size of the array (陣列長度) is determined by the number of values in parentheses
  - 註：陣列長度也取決於幾筆數值放入初始序列。

10.4.2 Initializing Arrays with Initializer Lists

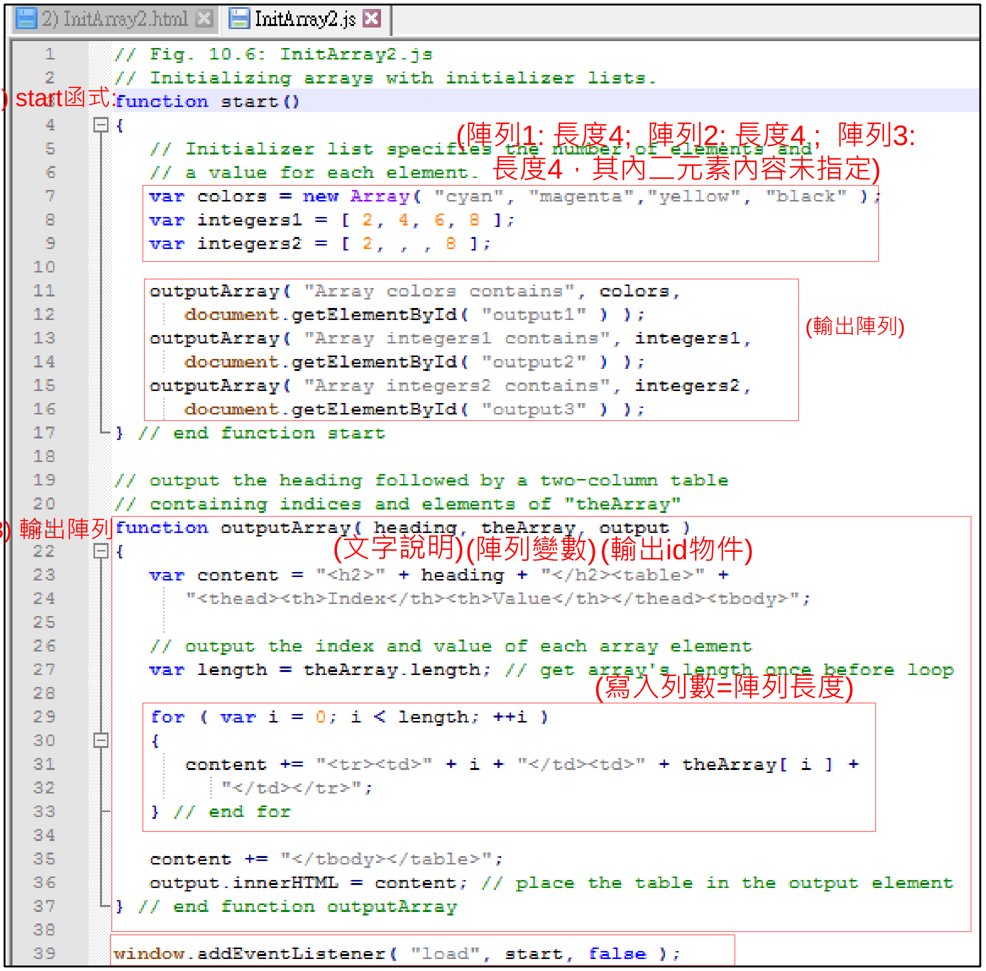
- Example:
  - 不同陣列宣告&初始化



```html
<!DOCTYPE html>

<!-- Fig. 10.5: InitArray2.html -->
<!-- Web page for showing the results of initializing arrays. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>Initializing an Array</title>
      <link rel = "stylesheet" type = "text/css" href = "tablestyle.css">
      <script src = "InitArray2.js"></script>
   </head>
   <body>
      <div id = "output1"></div>
      <div id = "output2"></div>
      <div id = "output3"></div>
   </body>
</html>
```

(JavaScript來源)

(3個輸出id)



```javascript
// Fig. 10.6: InitArray2.js
// Initializing arrays with initializer lists.
function start()
{
   // Initializer list specifies the number of elements and
   // a value for each element.
   var colors = new Array( "cyan", "magenta","yellow", "black" );
   var integers1 = [ 2, 4, 6, 8 ];
   var integers2 = [ 2, , , 8 ];

   outputArray( "Array colors contains", colors,
      document.getElementById( "output1" ) );
   outputArray( "Array integers1 contains", integers1,
      document.getElementById( "output2" ) );
   outputArray( "Array integers2 contains", integers2,
      document.getElementById( "output3" ) );
} // end function start

// output the heading followed by a two-column table
// containing indices and elements of "theArray"
function outputArray( heading, theArray, output )
{
   var content = "<h2>" + heading + "</h2><table>" +
      "<thead><th>Index</th><th>Value</th></thead><tbody>";

   // output the index and value of each array element
   var length = theArray.length; // get array's length once before loop

   for ( var i = 0; i < length; ++i )
   {
      content += "<tr><td>" + i + "</td><td>" + theArray[ i ] +
         "</td></tr>";
   } // end for

   content += "</tbody></table>";
   output.innerHTML = content; // place the table in the output element
} // end function outputArray

window.addEventListener( "load", start, false );
```

(2) start函式:

(陣列1: 長度4; 陣列2: 長度4; 陣列3: 長度4，其內二元素內容未指定)

(輸出陣列)

(3) 輸出陣列

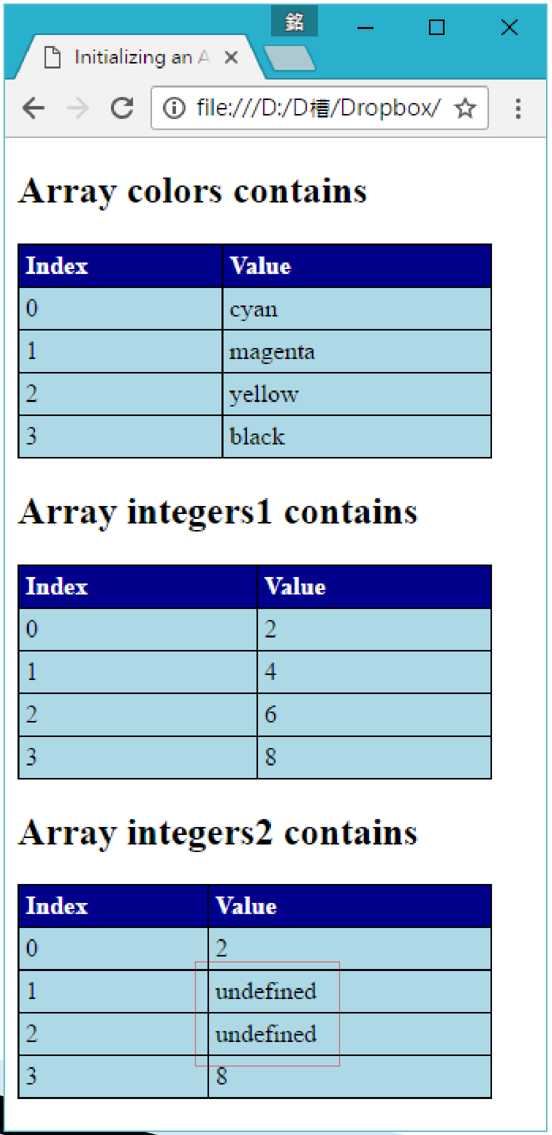
(文字說明)(陣列變數)(輸出id物件)

(寫入列數=陣列長度)

(1) 註冊event handler: 當load時，執行start函式

10.4.2 Initializing Arrays with Initializer Lists

(輸出)



Array colors contains

Index	Value
0	cyan
1	magenta
2	yellow
3	black

Array integers1 contains

Index	Value
0	2
1	4
2	6
3	8

Array integers2 contains

Index	Value
0	2
1	undefined
2	undefined
3	8

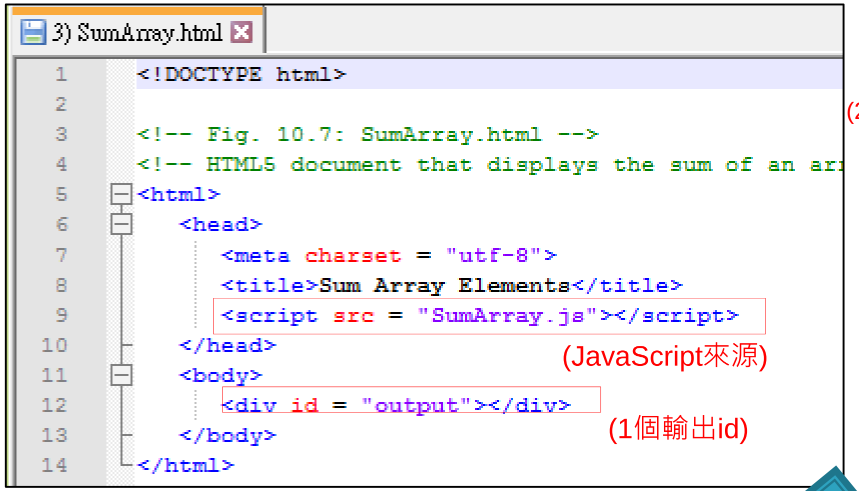
(陣列1: 長度4，4項初始值： cyan, magenta, yellow, black)

(陣列2: 長度4， 4項初始值： 2, 4, 6, 8)

(陣列3: 長度4， 2項初始值、2項未定義)

10.4.3 Summing the Elements of an Array with for and for…in

- Example: JavaScript’s for…in Repetition Statement
  - Enables a script to perform a task for each element in an array (迴圈根據陣列元素個數)

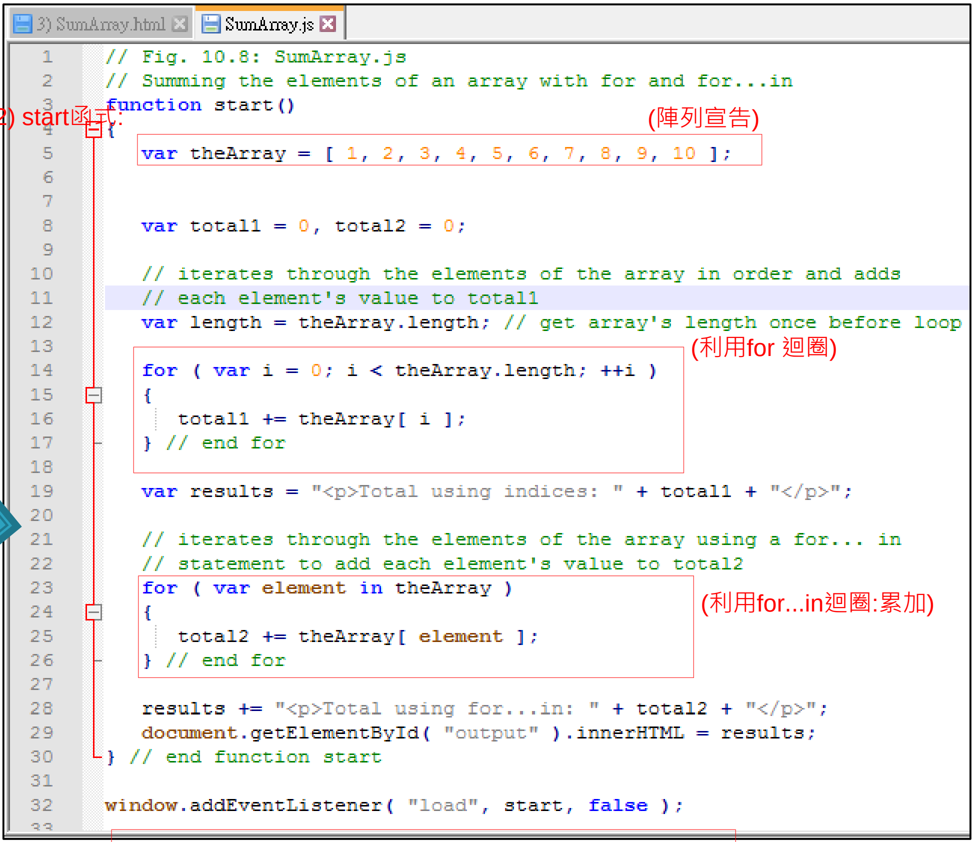


3) SumArray.html

```
<!DOCTYPE html>

<!-- Fig. 10.7: SumArray.html -->
<!-- HTML5 document that displays the sum of an ar
<html>
   <head>
      <meta charset = "utf-8">
      <title>Sum Array Elements</title>
      <script src = "SumArray.js"></script>
   </head>
   <body>
      <div id = "output"></div>
   </body>
</html>
```

(JavaScript來源)

(1個輸出id)



3) SumArray.html   SumArray.js

```
// Fig. 10.8: SumArray.js
// Summing the elements of an array with for and for...in
function start()
{
   var theArray = [ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 ];


   var total1 = 0, total2 = 0;

   // iterates through the elements of the array in order and adds
   // each element's value to total1
   var length = theArray.length; // get array's length once before loop

   for ( var i = 0; i < theArray.length; ++i )
   {
      total1 += theArray[ i ];
   } // end for

   var results = "<p>Total using indices: " + total1 + "</p>";

   // iterates through the elements of the array using a for... in
   // statement to add each element's value to total2
   for ( var element in theArray )
   {
      total2 += theArray[ element ];
   } // end for

   results += "<p>Total using for...in: " + total2 + "</p>";
   document.getElementById( "output" ).innerHTML = results;
} // end function start

window.addEventListener( "load", start, false );
```

(2) start函式:

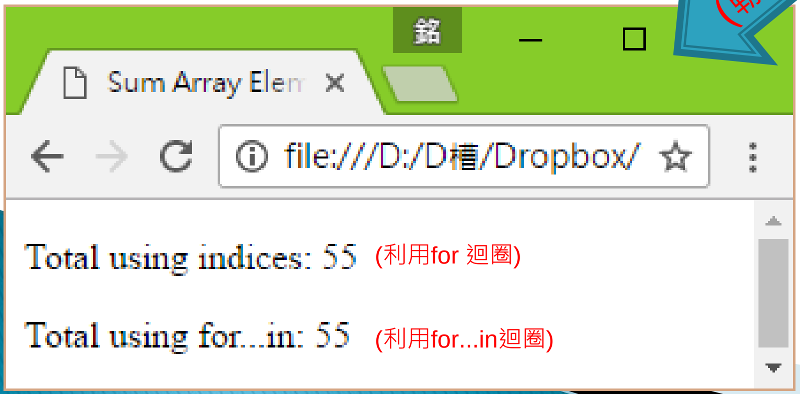
(陣列宣告)

(利用for 迴圈)

(利用for...in迴圈:累加)

(1) 註冊event handler: 當load時，執行start函式

(輸出)



Total using indices: 55 (利用for 迴圈)

Total using for...in: 55 (利用for...in迴圈)

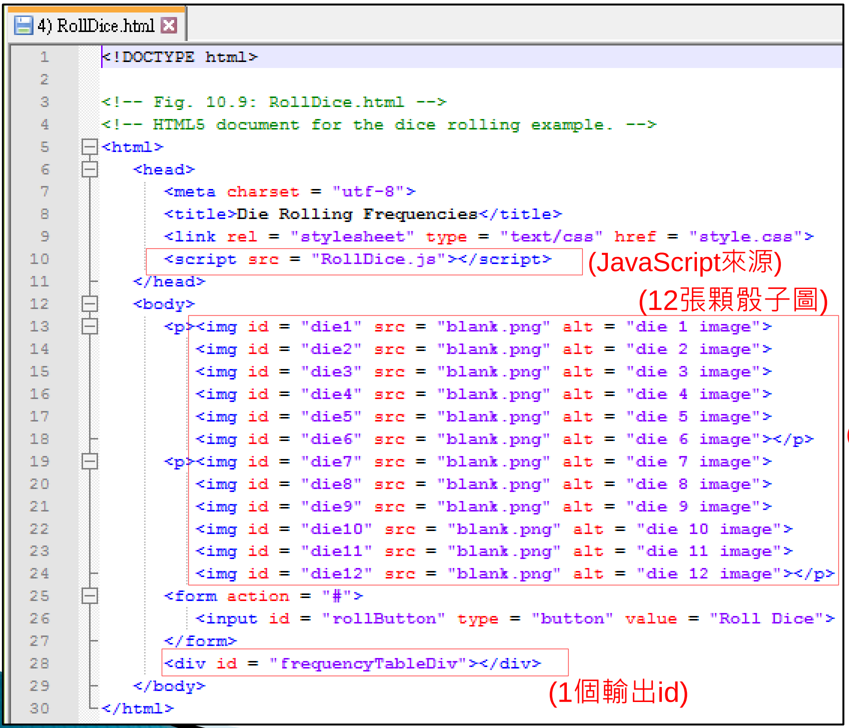
Question #1

- What is the output (輸出結果) if we change Line #5 in previous example to the following code?
  - var theArray = [1, 2, , , 5, 6, 7, 8, 9, 10];

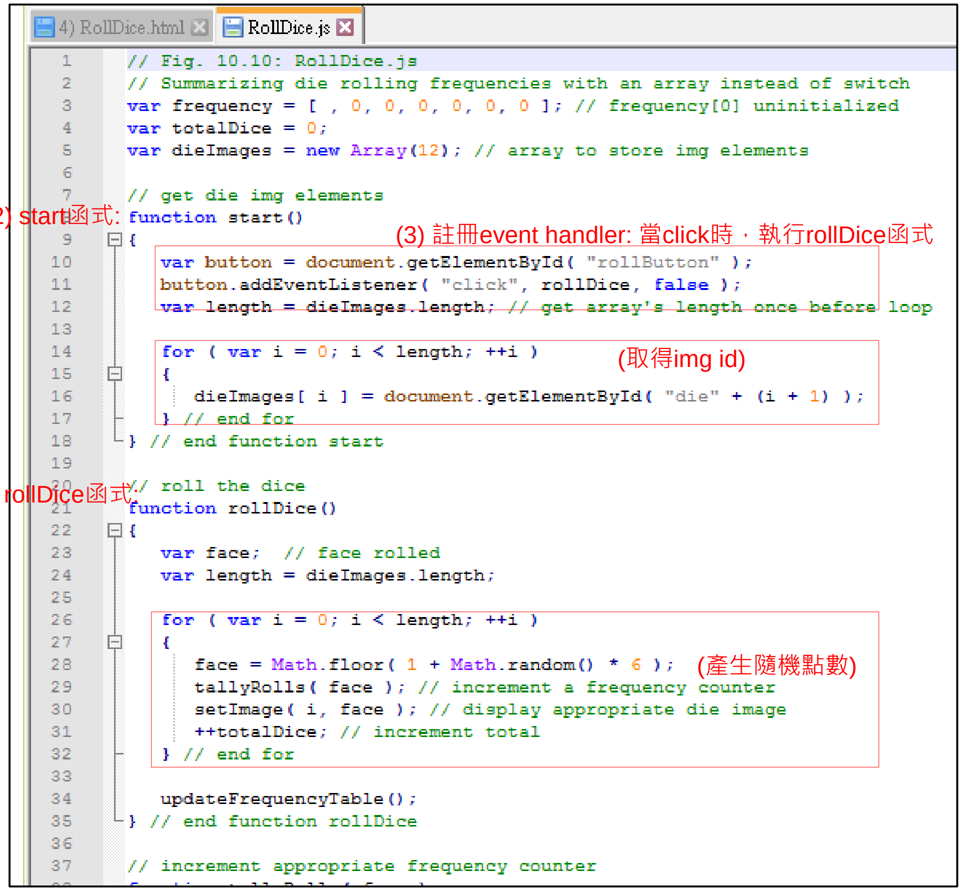
試試看!

10.4.4 Using the Elements of an Array as Counters

- Example: (陣列版本) To roll 12 dice at a time and kept statistics (統計) showing the number of times and the percentage (頻率) of the time each face occurred.

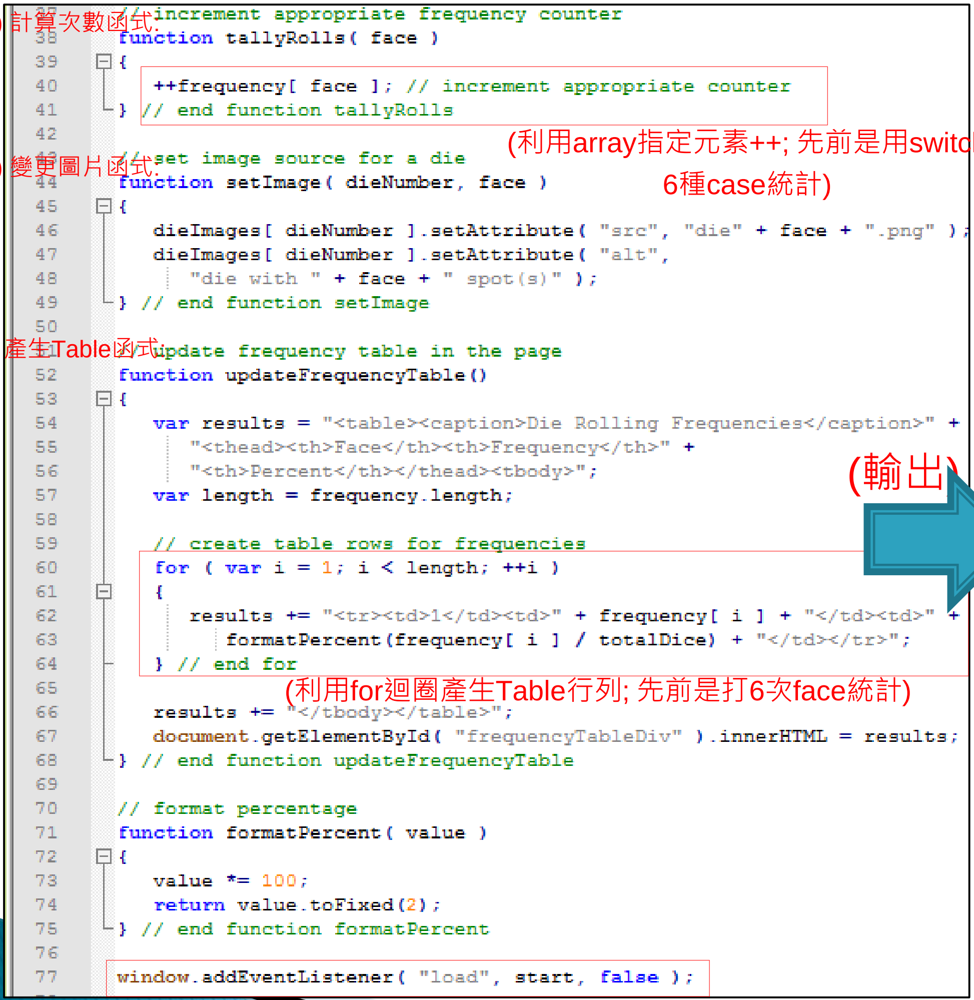


```html
<!DOCTYPE html>

<!-- Fig. 10.9: RollDice.html -->
<!-- HTML5 document for the dice rolling example. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>Die Rolling Frequencies</title>
      <link rel = "stylesheet" type = "text/css" href = "style.css">
      <script src = "RollDice.js"></script>
   </head>
   <body>
      <p><img id = "die1" src = "blank.png" alt = "die 1 image">
         <img id = "die2" src = "blank.png" alt = "die 2 image">
         <img id = "die3" src = "blank.png" alt = "die 3 image">
         <img id = "die4" src = "blank.png" alt = "die 4 image">
         <img id = "die5" src = "blank.png" alt = "die 5 image">
         <img id = "die6" src = "blank.png" alt = "die 6 image"></p>
      <p><img id = "die7" src = "blank.png" alt = "die 7 image">
         <img id = "die8" src = "blank.png" alt = "die 8 image">
         <img id = "die9" src = "blank.png" alt = "die 9 image">
         <img id = "die10" src = "blank.png" alt = "die 10 image">
         <img id = "die11" src = "blank.png" alt = "die 11 image">
         <img id = "die12" src = "blank.png" alt = "die 12 image"></p>
      <form action = "#">
         <input id = "rollButton" type = "button" value = "Roll Dice">
      </form>
      <div id = "frequencyTableDiv"></div>
   </body>
</html>
```



```javascript
// Fig. 10.10: RollDice.js
// Summarizing die rolling frequencies with an array instead of switch
var frequency = [ , 0, 0, 0, 0, 0, 0 ]; // frequency[0] uninitialized
var totalDice = 0;
var dieImages = new Array(12); // array to store img elements

// get die img elements
function start()
{
   var button = document.getElementById( "rollButton" );
   button.addEventListener( "click", rollDice, false );
   var length = dieImages.length; // get array's length once before loop

   for ( var i = 0; i < length; ++i )
   {
      dieImages[ i ] = document.getElementById( "die" + (i + 1) );
   } // end for
} // end function start

// roll the dice
function rollDice()
{
   var face;  // face rolled
   var length = dieImages.length;

   for ( var i = 0; i < length; ++i )
   {
      face = Math.floor( 1 + Math.random() * 6 );
      tallyRolls( face ); // increment a frequency counter
      setImage( i, face ); // display appropriate die image
      ++totalDice; // increment total
   } // end for

   updateFrequencyTable();
} // end function rollDice

// increment appropriate frequency counter
```

10.4.4 Using the Elements of an Array as Counters



(4) 計算次數函式:

```
// increment appropriate frequency counter
function tallyRolls( face )
{
   ++frequency[ face ]; // increment appropriate counter
} // end function tallyRolls
```

(利用array指定元素++; 先前是用switch 6種case統計)

(5) 變更圖片函式:

```
// set image source for a die
function setImage( dieNumber, face )
{
   dieImages[ dieNumber ].setAttribute( "src", "die" + face + ".png" );
   dieImages[ dieNumber ].setAttribute( "alt",
      "die with " + face + " spot(s)" );
} // end function setImage
```

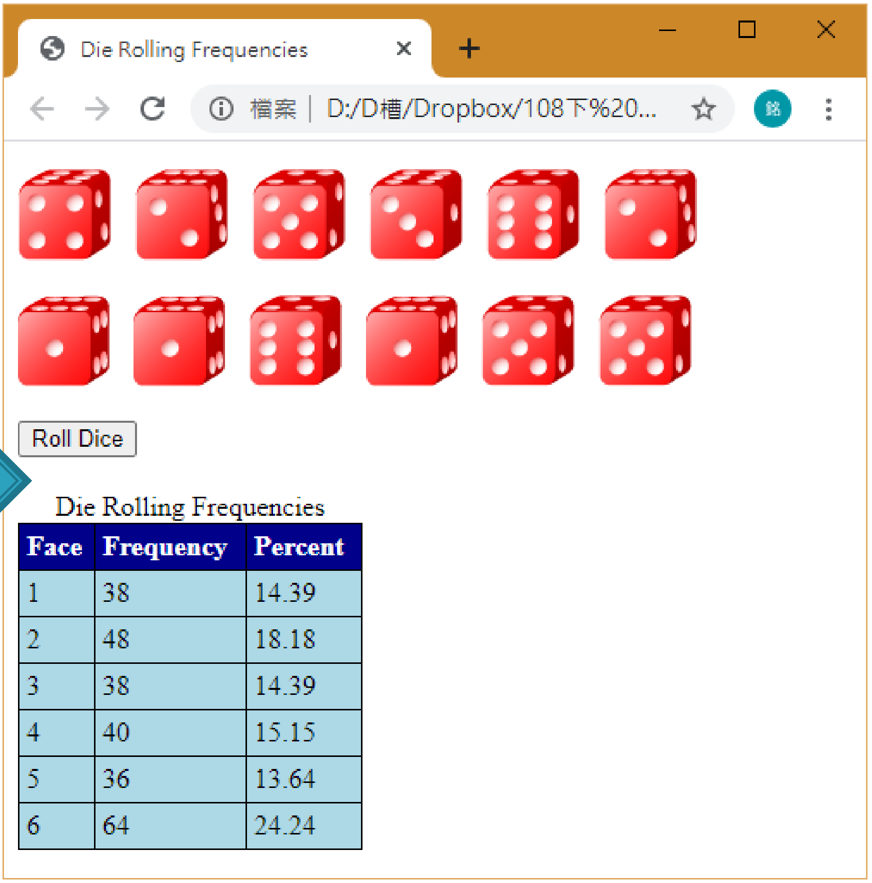
(6) 產生Table函式:

```
// update frequency table in the page
function updateFrequencyTable()
{
   var results = "<table><caption>Die Rolling Frequencies</caption>" +
      "<thead><th>Face</th><th>Frequency</th>" +
      "<th>Percent</th></thead><tbody>";
   var length = frequency.length;

   // create table rows for frequencies
   for ( var i = 1; i < length; ++i )
   {
      results += "<tr><td>1</td><td>" + frequency[ i ] + "</td><td>" +
         formatPercent(frequency[ i ] / totalDice) + "</td></tr>";
   } // end for

   results += "</tbody></table>";
   document.getElementById( "frequencyTableDiv" ).innerHTML = results;
} // end function updateFrequencyTable

// format percentage
function formatPercent( value )
{
   value *= 100;
   return value.toFixed(2);
} // end function formatPercent

window.addEventListener( "load", start, false );
```

(利用for迴圈產生Table行列; 先前是打6次face統計)

(1) 註冊event handler: 當load時，執行start函式

(輸出)

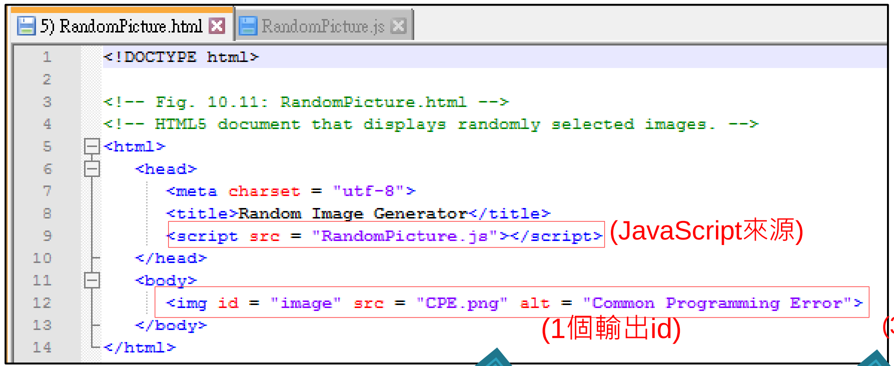


Die Rolling Frequencies

Face	Frequency	Percent
1	38	14.39
2	48	18.18
3	38	14.39
4	40	15.15
5	36	13.64
6	64	24.24

10.5 Random Image Generator Using Arrays

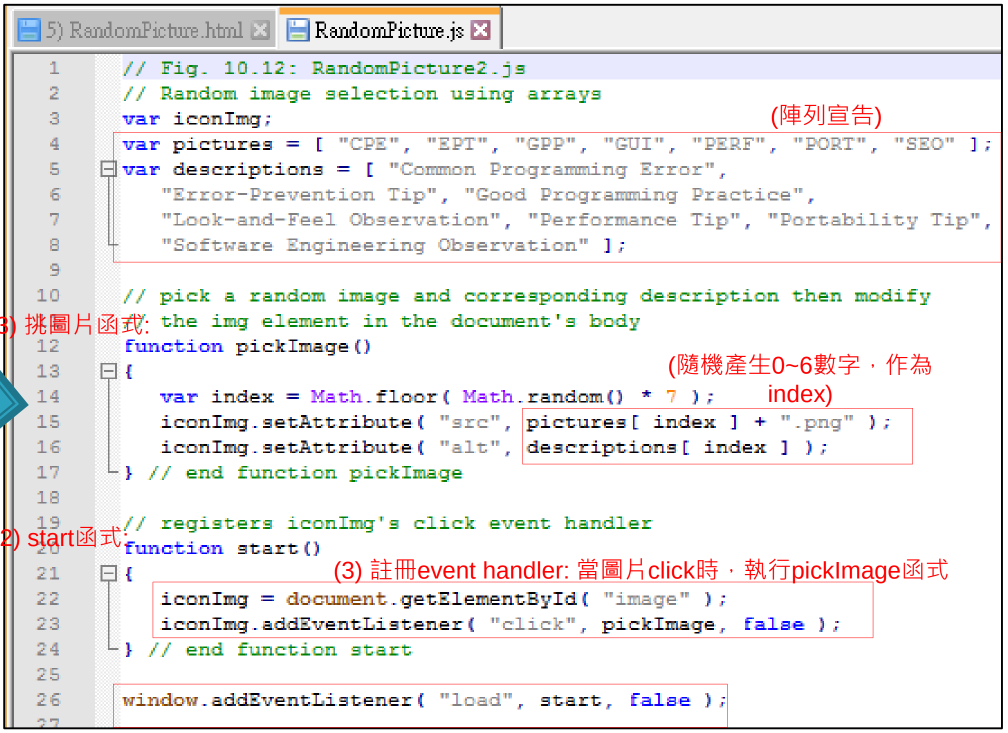
- Example: (以陣列方式) the random image generator required image files to be named with the word die followed by a number from 1 to 6 and the file extension .png (e.g, die1.png).



```
<!DOCTYPE html>

<!-- Fig. 10.11: RandomPicture.html -->
<!-- HTML5 document that displays randomly selected images. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>Random Image Generator</title>
      <script src = "RandomPicture.js"></script>
   </head>
   <body>
      <img id = "image" src = "CPE.png" alt = "Common Programming Error">
   </body>
</html>
```

(JavaScript來源)

(1個輸出id)



```
// Fig. 10.12: RandomPicture2.js
// Random image selection using arrays
var iconImg;
var pictures = [ "CPE", "EPT", "GPP", "GUI", "PERF", "PORT", "SEO" ];
var descriptions = [ "Common Programming Error",
   "Error-Prevention Tip", "Good Programming Practice",
   "Look-and-Feel Observation", "Performance Tip", "Portability Tip",
   "Software Engineering Observation" ];

// pick a random image and corresponding description then modify
// the img element in the document's body
function pickImage()
{
   var index = Math.floor( Math.random() * 7 );
   iconImg.setAttribute( "src", pictures[ index ] + ".png" );
   iconImg.setAttribute( "alt", descriptions[ index ] );
} // end function pickImage

// registers iconImg's click event handler
function start()
{
   iconImg = document.getElementById( "image" );
   iconImg.addEventListener( "click", pickImage, false );
} // end function start

window.addEventListener( "load", start, false );
```

(陣列宣告)

(3) 挑圖片函式

(隨機產生0~6數字，作為index)

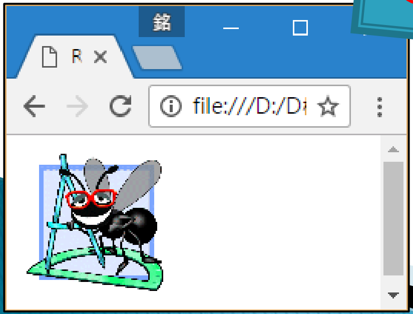
(2) start函式

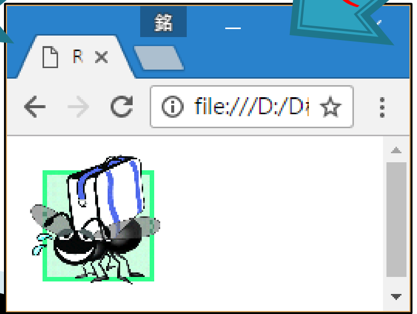
(3) 註冊event handler: 當圖片click時，執行pickImage函式

(1) 註冊event handler: 當load時，執行start函式

(輸出)

(輸出)





10.6 References and Reference Parameters

兩種傳遞參數至函式(或方法)

- Two ways to pass arguments to functions (or methods)
  - pass-by-value (傳值)
  - pass-by-reference (傳參考)

- (1) Pass-by-value (傳值)
  - A *copy* of the argument's value (複制參數值) is made and is passed to the called function
    - 註：傳遞複製的參數值給函式
  - In JavaScript, numbers (數字), boolean values (布林值) and strings (字串) are passed to functions by value.
    - 註：在Javascript中，數字、布林值及字串型態都只是傳值(pass-by-value).

- (2) Pass-by-reference (傳參考)
  - The caller allows the called function to *modify* the data
    - 註：呼叫者允許函數修改傳遞變數之內容。
    - 註：可提高存取效率，由於能避免複製大量的資訊，缺點是安全性較低
  - All objects are passed to functions by reference
    - 註：在JavaScript中，所有object傳遞都是傳參考(pass-by-reference).

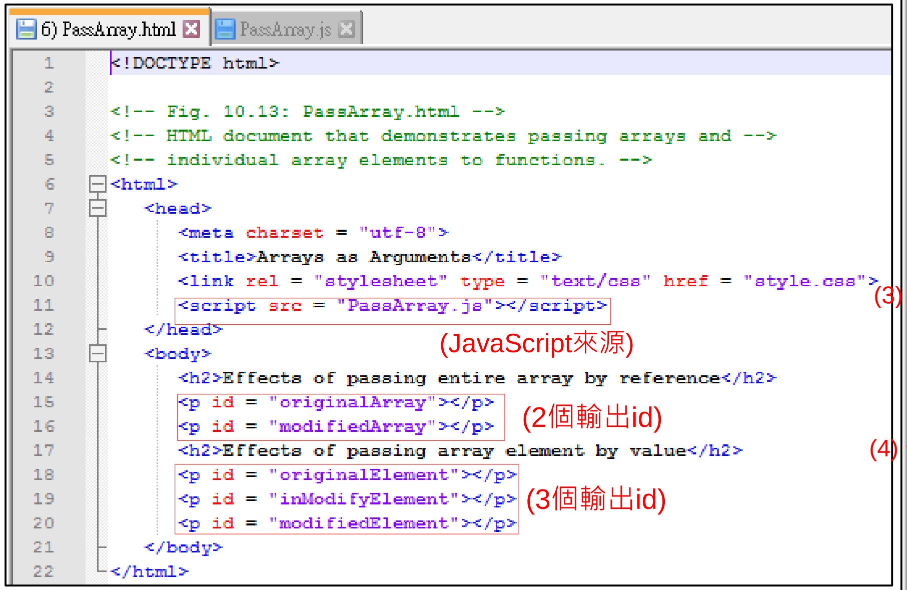
10.7 Passing Arrays to Functions

functionname (arrayname); 傳遞陣式至函式

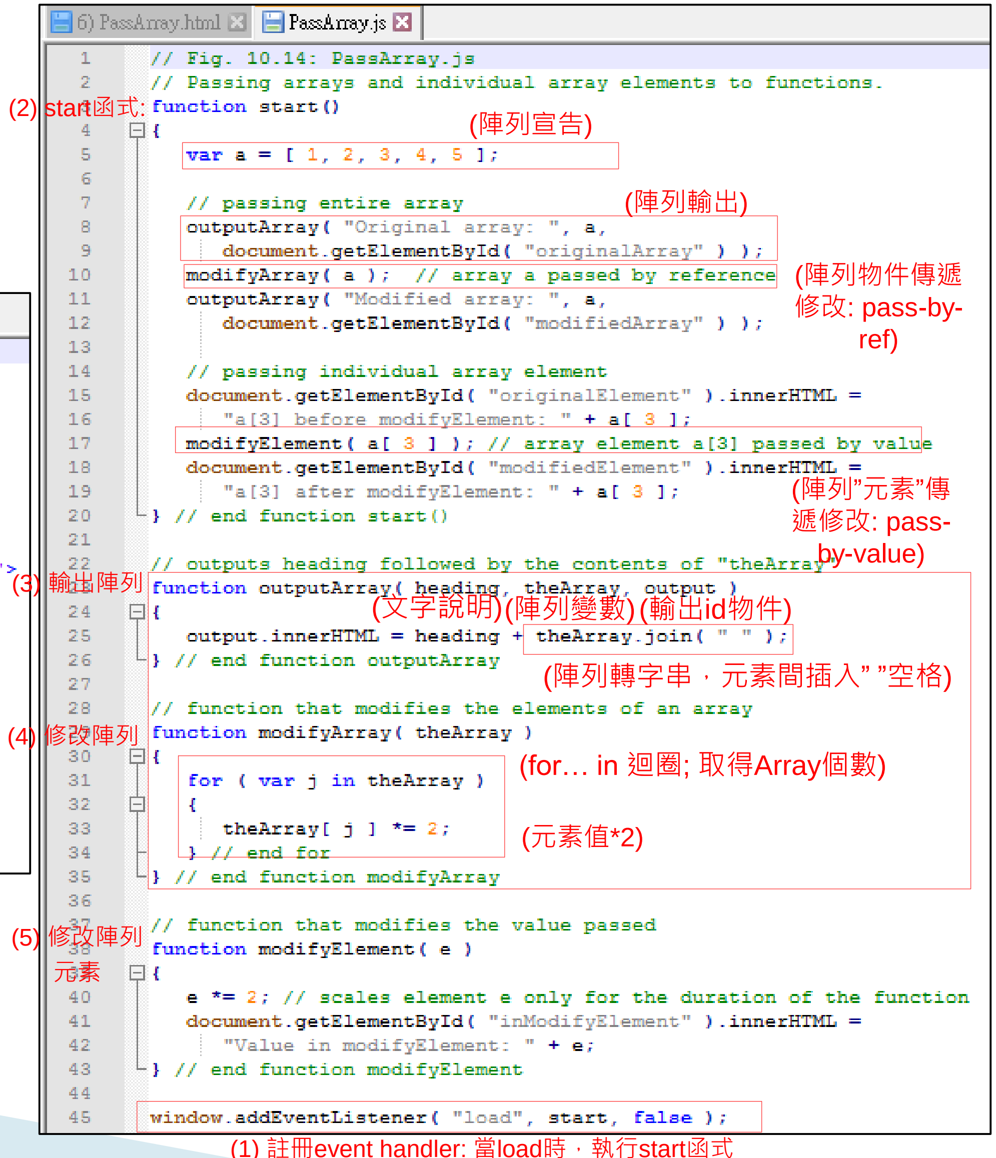
- Pass an array as an argument to a function
  - Specify the array's name without brackets
    - 註：傳遞陣式給函式時，使用陣列名稱即可 (不用加括號)
- Although entire arrays are passed by reference, *individual numeric and boolean array elements* are passed *by value* exactly as simple numeric and boolean variables are passed
  - 註：傳遞單一的陣列元素，則是傳值(passed-*by-value*)
  - 註：這類單一的資料稱為scalars or scalar quantities (純量)
  - 註：使用方式 *arrayname[元素編號]*
- *arrayname*.join("…"); 陣列轉字串，並在元素間插入"間隔"
  - Returns a string that contains all of the elements of an array, separated by the string supplied in the function's argument
    - 註：將陣列轉成字串，並在陣列元素間插入"指定間隔"，ex: " "空白、","逗號
    - 註：若未指定間隔，則會插入"空字串"(empty string)

10.7 Passing Arrays to Functions (Cont.)

- Example:

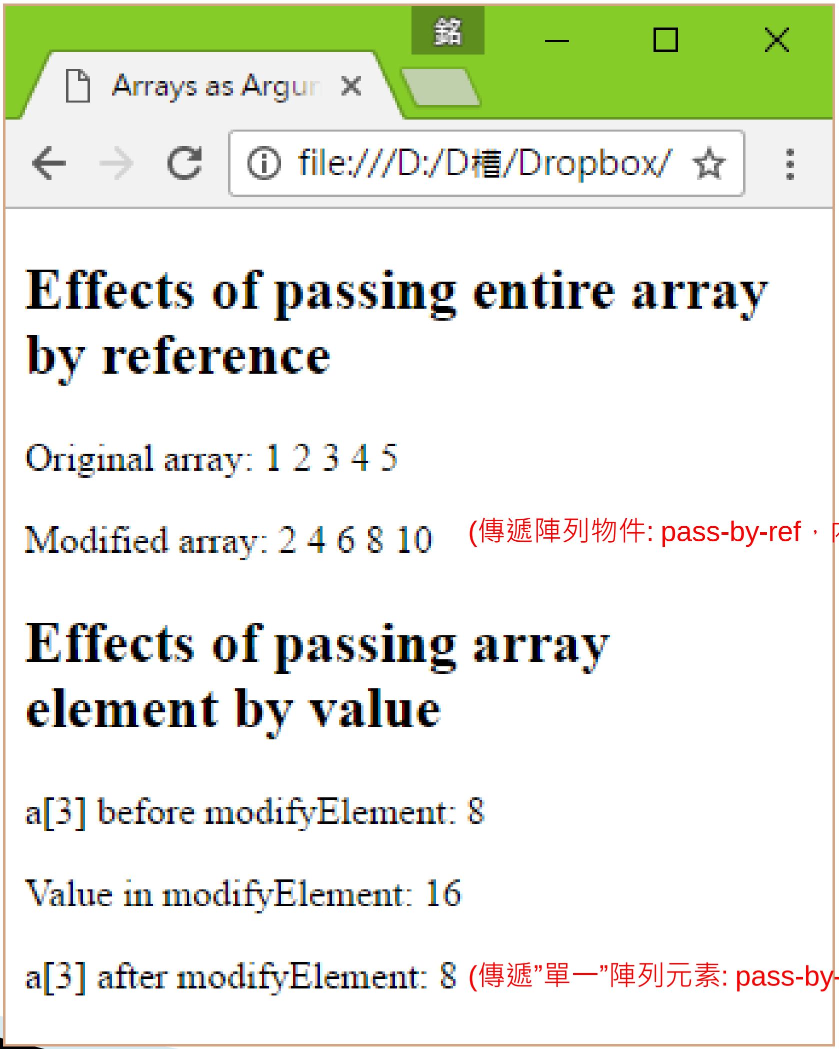


```html
<!DOCTYPE html>

<!-- Fig. 10.13: PassArray.html -->
<!-- HTML document that demonstrates passing arrays and -->
<!-- individual array elements to functions. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>Arrays as Arguments</title>
      <link rel = "stylesheet" type = "text/css" href = "style.css">
      <script src = "PassArray.js"></script>
   </head>
   <body>
      <h2>Effects of passing entire array by reference</h2>
      <p id = "originalArray"></p>
      <p id = "modifiedArray"></p>
      <h2>Effects of passing array element by value</h2>
      <p id = "originalElement"></p>
      <p id = "inModifyElement"></p>
      <p id = "modifiedElement"></p>
   </body>
</html>
```


6) PassArray.html
PassArray.js
(2) start函式:
(陣列宣告)
(陣列輸出)
(陣列物件傳遞修改: pass-by-ref)
(陣列"元素"傳遞修改: pass-by-value)
(3) 輸出陣列
(文字說明)(陣列變數)(輸出id物件)
(陣列轉字串，元素間插入" "空格)
(4) 修改陣列
(for… in 迴圈; 取得Array個數)
(元素值*2)
(5) 修改陣列元素
(1) 註冊event handler: 當load時，執行start函式

```javascript
// Fig. 10.14: PassArray.js
// Passing arrays and individual array elements to functions.
function start()
{
   var a = [ 1, 2, 3, 4, 5 ];

   // passing entire array
   outputArray( "Original array: ", a,
      document.getElementById( "originalArray" ) );
   modifyArray( a );  // array a passed by reference
   outputArray( "Modified array: ", a,
      document.getElementById( "modifiedArray" ) );

   // passing individual array element
   document.getElementById( "originalElement" ).innerHTML =
      "a[3] before modifyElement: " + a[ 3 ];
   modifyElement( a[ 3 ] ); // array element a[3] passed by value
   document.getElementById( "modifiedElement" ).innerHTML =
      "a[3] after modifyElement: " + a[ 3 ];
} // end function start()

// outputs heading followed by the contents of "theArray"
function outputArray( heading, theArray, output )
{
   output.innerHTML = heading + theArray.join( " " );
} // end function outputArray

// function that modifies the elements of an array
function modifyArray( theArray )
{
   for ( var j in theArray )
   {
      theArray[ j ] *= 2;
   } // end for
} // end function modifyArray

// function that modifies the value passed
function modifyElement( e )
{
   e *= 2; // scales element e only for the duration of the function
   document.getElementById( "inModifyElement" ).innerHTML =
      "Value in modifyElement: " + e;
} // end function modifyElement

window.addEventListener( "load", start, false );
```

10.4.3 Summing the Elements of an Array with for and for…in



```
Effects of passing entire array by reference

Original array: 1 2 3 4 5

Modified array: 2 4 6 8 10

Effects of passing array element by value

a[3] before modifyElement: 8

Value in modifyElement: 16

a[3] after modifyElement: 8
```

(輸出)

(傳遞陣列物件: pass-by-ref，內容值會被修改)

(傳遞"單一"陣列元素: pass-by-value，真正的陣列值不會被修改)

10.9 Searching Arrays with Array Method `indexOf`

Searching Arrays 陣列搜尋

- To determine whether an array contains a value that matches a certain *key value (指定數值)*.
  - 註：判斷此陣列是否包含一指定數值

- The process of locating a particular element value in an array is called *searching*.
  - 註："陣列搜尋"指的找出陣列中某個元素的"位置"。

arrayname.indexOf(搜尋值); 由"前往後"搜尋指定值

arrayname.lastindexOf(搜尋值); 由"後往前"搜尋指定值

- The built-in (內建) methods `indexOf` and `lastIndexOf` for searching arrays.
  - Method `indexOf` searches for the first occurrence (由前往後找) of the specified key value
    - 註：由" 前往後" 搜尋
  - Method `lastIndexOf` searches for the last occurrence (由後往前找) of the specified key value.
    - 註：由" 後往前" 搜尋
- If the key value is found in the array, each method returns the index (回傳索引值) of that value; otherwise, -1 is returned.
  - 註：key value 搜尋到時，回傳該元素索引值，否則回傳-1

10.9 Searching Arrays with Array Method `indexOf` (Cont.)

inputelement.value 取得某input值內容

- Every `input element` has a **`value` property** that can be used to get (讀取) or set (寫入) the element's value.
  - 註：取得網頁input element的value內容

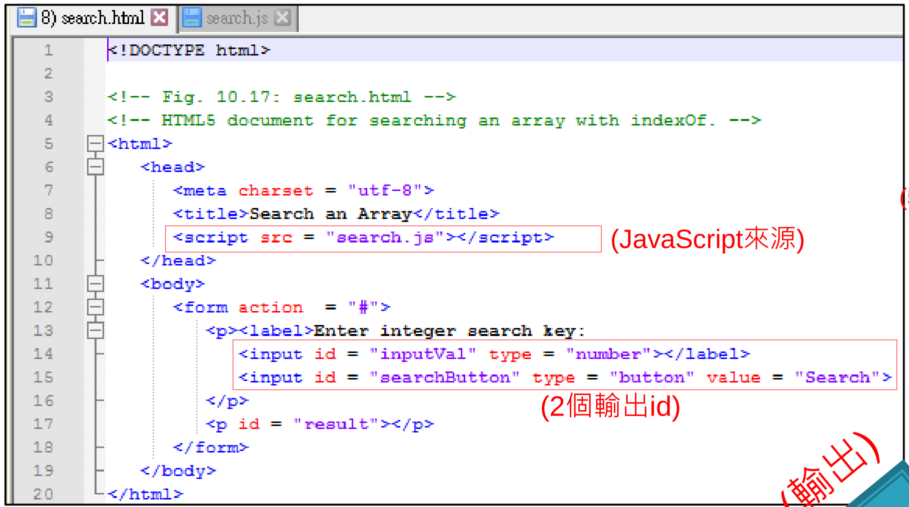
`arrayname.indexOf(搜尋值, 起始位置);    //指定位置開始搜尋`

`arrayname.lastIndexOf(搜尋值, 起始位置);`

- An optional second argument to methods `indexOf` and `lastIndexOf` that represents the index from which to start (從何處開始搜尋) the search.
  - 註：第二欄參數是指定搜尋起始位置
  - 註：預設不填，啟始位置為從0位置開始
  - 註：若填入的啟始位置超出陣列長度，則回傳-1
  - 註：若填入的啟始位置為負值，則為倒數 (ex: -1 代表"倒數第一個位置")

10.9 Searching Arrays with Array Method indexOf (Cont.)

- Example:
  - 搜尋陣列之索引值

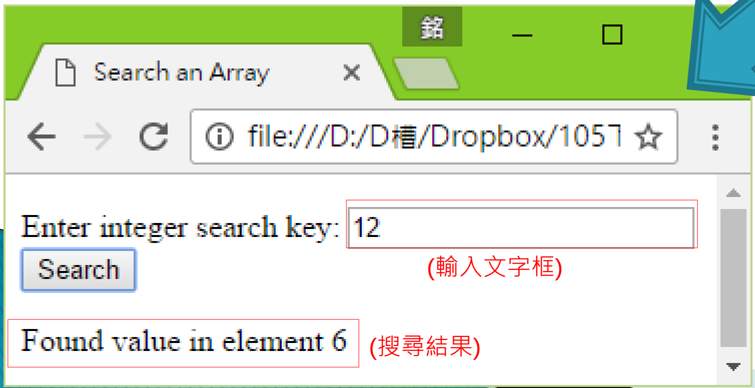


```
<!DOCTYPE html>

<!-- Fig. 10.17: search.html -->
<!-- HTML5 document for searching an array with indexOf. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>Search an Array</title>
      <script src = "search.js"></script>
   </head>
   <body>
      <form action  = "#">
         <p><label>Enter integer search key:
            <input id = "inputVal" type = "number"></label>
            <input id = "searchButton" type = "button" value = "Search">
         </p>
         <p id = "result"></p>
      </form>
   </body>
</html>
```

(JavaScript來源)

(2個輸出id)

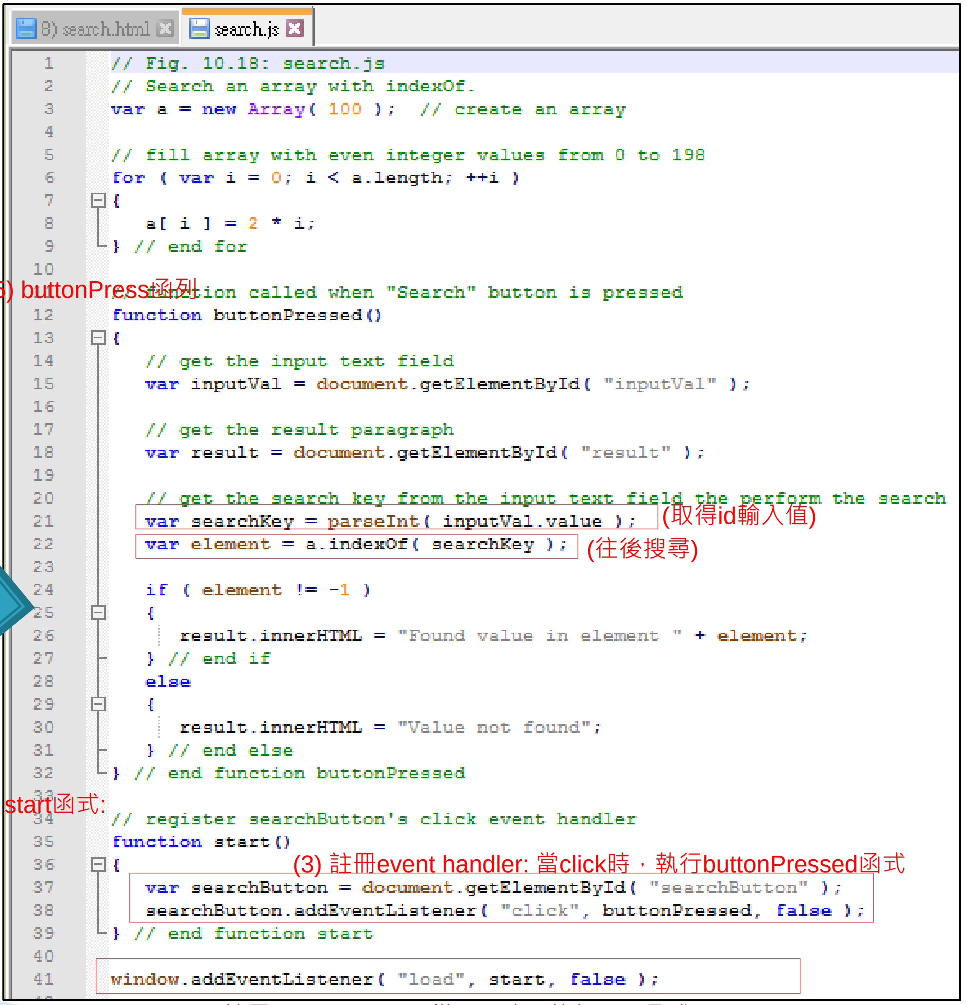
(輸出)



```
Enter integer search key: 12
Search

Found value in element 6
```

(輸入文字框)

(搜尋結果)



```
// Fig. 10.18: search.js
// Search an array with indexOf.
var a = new Array( 100 );  // create an array

// fill array with even integer values from 0 to 198
for ( var i = 0; i < a.length; ++i )
{
   a[ i ] = 2 * i;
} // end for

// function called when "Search" button is pressed
function buttonPressed()
{
   // get the input text field
   var inputVal = document.getElementById( "inputVal" );

   // get the result paragraph
   var result = document.getElementById( "result" );

   // get the search key from the input text field the perform the search
   var searchKey = parseInt( inputVal.value );
   var element = a.indexOf( searchKey );

   if ( element != -1 )
   {
      result.innerHTML = "Found value in element " + element;
   } // end if
   else
   {
      result.innerHTML = "Value not found";
   } // end else
} // end function buttonPressed

// register searchButton's click event handler
function start()
{
   var searchButton = document.getElementById( "searchButton" );
   searchButton.addEventListener( "click", buttonPressed, false );
} // end function start

window.addEventListener( "load", start, false );
```

(5) buttonPress函列

(取得id輸入值)

(往後搜尋)

(2) start函式:

(3) 註冊event handler: 當click時，執行buttonPressed函式

(1) 註冊event handler: 當load時，執行start函式

10.10 Multidimensional Arrays

2維陣列

- To identify a two-dimensional (2維) or multidimensional array (多維陣列) element
  - By specifying the two indices (2層array)
    - 註：透過2層array方式宣告
  - By convention, the first identifies (第一層) the element’s row, and the second identifies (第二層) the element’s column
    - 註：第一層是宣告陣列的”列元素”，第二層是宣告陣列的”行元素”

m-by-n array (m x n 維陣列)

- An array with *m* rows (列) and *n* columns (行) is called an *m*-by-*n* array (m x n 維陣列)

arrayname[列索引][行索引]; 2維陣列元素的存取

- Two-dimensional array element accessed using an element name of the form *`arrayname[ row ][ column ]`*
  - 註：列索引及行索引是用來識別指定元素

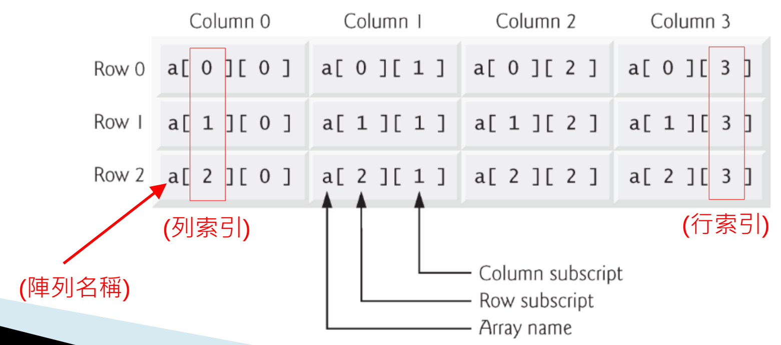


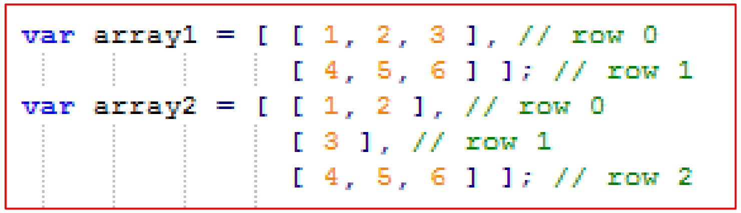
Fig. 10.19 | [illegible]ensional array with three rows and four

10.10 Multidimensional Arrays (Cont.)

arrayname[[…], […], …]; 多維陣列宣告

- Multidimensional arrays can be initialized (初始化) in declarations like a one-dimensional array (如同一維陣列宣告), with values grouped by row (再插入子陣列) in square brackets
  - 註：Javascript多維陣列宣告，如同一陣列宣告，在列的欄位內再插入子陣列。
  - 註：瀏覽器會去每一列計算有多少行元素。

- A multidimensional array (多維陣列) in which each row has a different number of columns (不同行元素個數) can be allocated dynamically with operator **new**
  - 註：多維陣列的行元素個數可以不同。
  - 如：

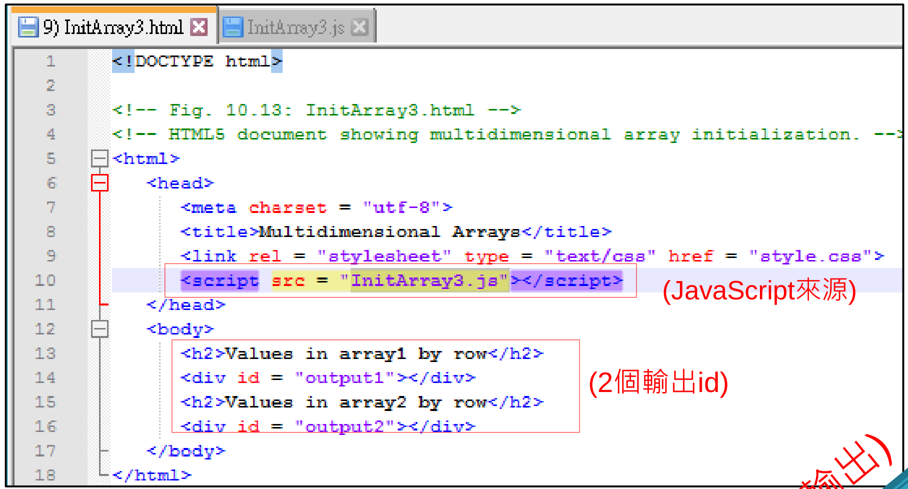


```
var array1 = [ [ 1, 2, 3 ], // row 0
               [ 4, 5, 6 ] ]; // row 1
var array2 = [ [ 1, 2 ], // row 0
               [ 3 ], // row 1
               [ 4, 5, 6 ] ]; // row 2
```

(不同行元素個數)

10.10 Multidimensional Arrays (Cont.)

- Example: 二維陣列



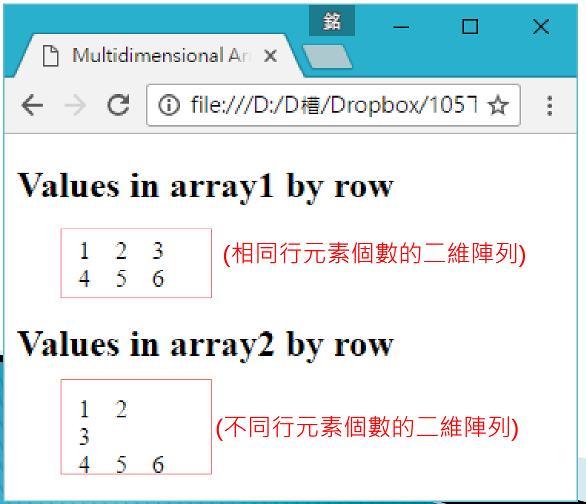
9) InitArray3.html | InitArray3.js

```
<!DOCTYPE html>

<!-- Fig. 10.13: InitArray3.html -->
<!-- HTML5 document showing multidimensional array initialization. -->
<html>
   <head>
      <meta charset = "utf-8">
      <title>Multidimensional Arrays</title>
      <link rel = "stylesheet" type = "text/css" href = "style.css">
      <script src = "InitArray3.js"></script>
   </head>
   <body>
      <h2>Values in array1 by row</h2>
      <div id = "output1"></div>
      <h2>Values in array2 by row</h2>
      <div id = "output2"></div>
   </body>
</html>
```

(JavaScript來源)

(2個輸出id)

(輸出)



Multidimensional Ar

file:///D:/D槽/Dropbox/1057

Values in array1 by row

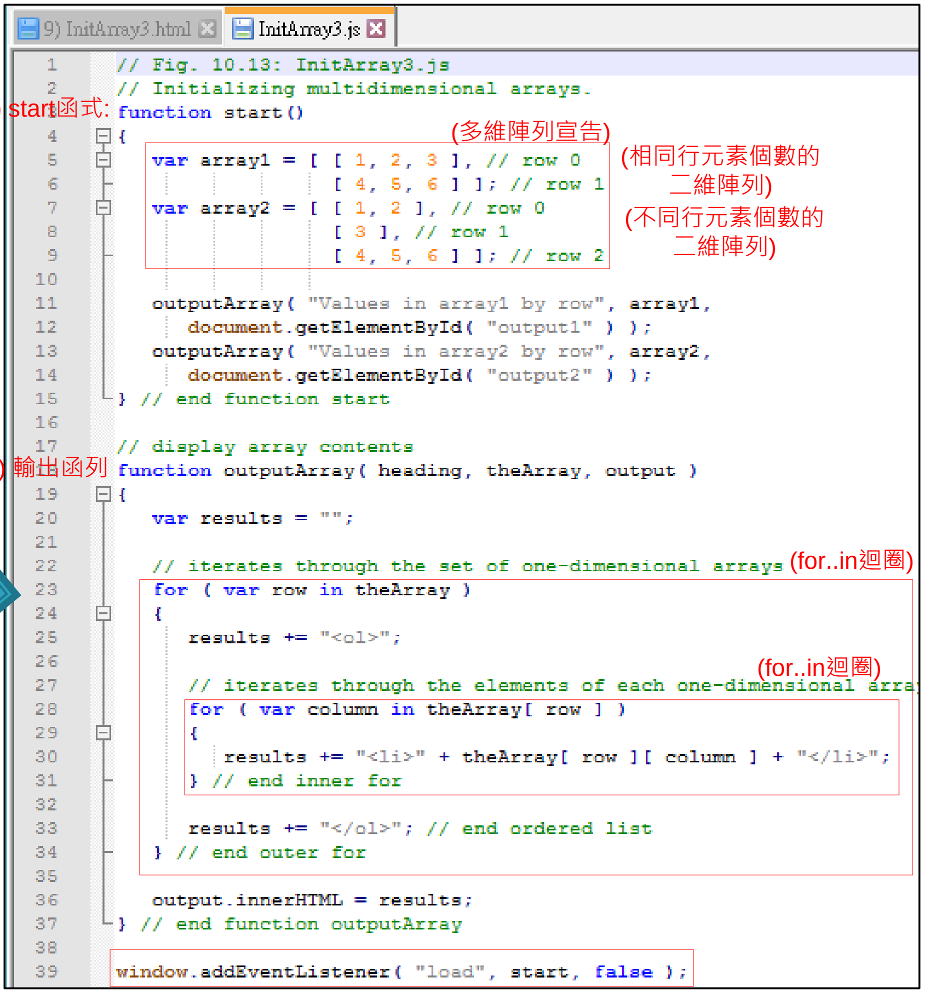
```
1  2  3
4  5  6
```

(相同行元素個數的二維陣列)

Values in array2 by row

```
1  2
3
4  5  6
```

(不同行元素個數的二維陣列)



9) InitArray3.html | InitArray3.js

```
// Fig. 10.13: InitArray3.js
// Initializing multidimensional arrays.
function start()
{
   var array1 = [ [ 1, 2, 3 ], // row 0
                  [ 4, 5, 6 ] ]; // row 1
   var array2 = [ [ 1, 2 ], // row 0
                  [ 3 ], // row 1
                  [ 4, 5, 6 ] ]; // row 2

   outputArray( "Values in array1 by row", array1,
      document.getElementById( "output1" ) );
   outputArray( "Values in array2 by row", array2,
      document.getElementById( "output2" ) );
} // end function start

// display array contents
function outputArray( heading, theArray, output )
{
   var results = "";

   // iterates through the set of one-dimensional arrays
   for ( var row in theArray )
   {
      results += "<ol>";

      // iterates through the elements of each one-dimensional arra
      for ( var column in theArray[ row ] )
      {
         results += "<li>" + theArray[ row ][ column ] + "</li>";
      } // end inner for

      results += "</ol>"; // end ordered list
   } // end outer for

   output.innerHTML = results;
} // end function outputArray

window.addEventListener( "load", start, false );
```

(2) start函式:

(多維陣列宣告)

(相同行元素個數的二維陣列)

(不同行元素個數的二維陣列)

(5) 輸出函列

(for..in迴圈)

(for..in迴圈)

(1) 註冊event handler: 當load時，執行start函式

Lab
(上機練習)

Question #1

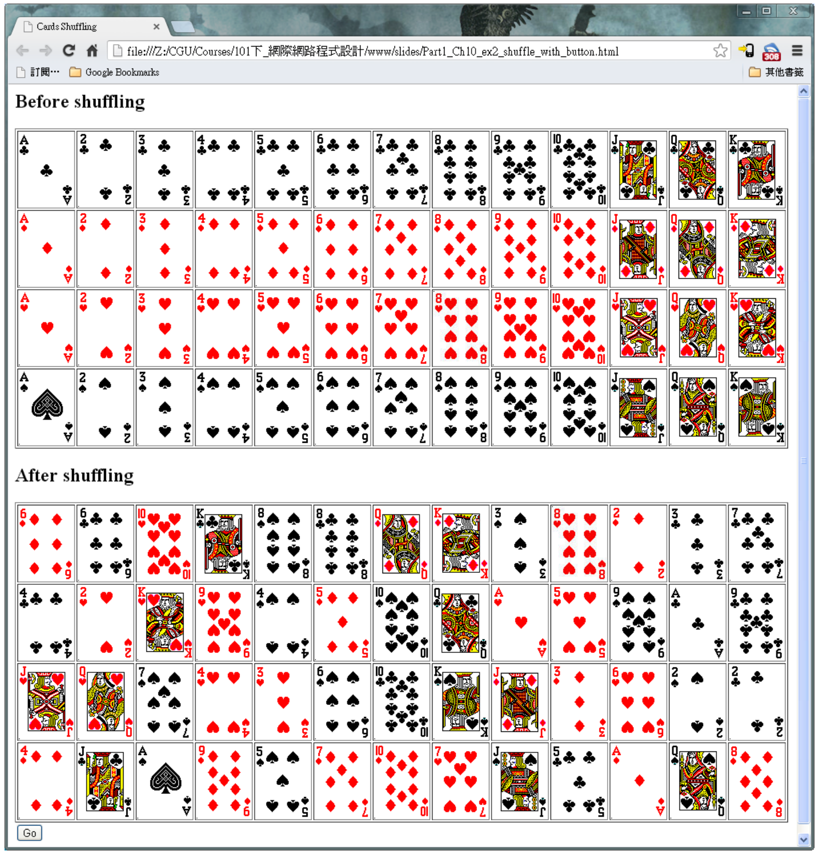
- What is the output (輸出結果) if we change Line #5 in previous example to the following code?
  - var theArray = [1, 2, , , 5, 6, 7, 8, 9, 10];

In-Class Exercise #1

- Modify the JavaScript program you wrote in Ch. 9 to shuffle (洗牌) a deck of cards and produce the following tables by array.
- 課中練習：
  - 請嘗試修改Lab5-2，達成洗牌重作 (花色數字不能重覆)。
    - 可使用陣列方式

- 注意：

  23:59 前上傳到ftp (否則視為遲交)：
  ftp://webdesign:webdesign@120.126.16.67:55555
  - Lab08
  - 請以"學號"建立資料匣 (否則扣分)


Cards Shuffling
file:///Z:/CGU/Courses/101下_網際網路程式設計/www/slides/Part1_Ch10_ex2_shuffle_with_button.html
訂閱…
Google Bookmarks
其他書籤
Go

Before shuffling

After shuffling

公告：期末專題Demo報告 (總成績35%+)

- Demo日期：
  - 第17週 (6/24)及 第18週 (7/1) (**註**：由於分組報告delay)
- 報告規則：
  - 每組報告時間：15分鐘 (含Q&A)
  - 每組報告順序，依先前分組報告順序，公佈於：
    - https://docs.google.com/spreadsheets/d/1lCve_2rLigQKjC85pGJUttPQ3w8PAHBbBA2GGFTQrJA/edit?usp=sharing
- Demo報告評分標準：
  1. 專題創意及實用性(22.5%)
  2. 技術困難性(22.5%)
  3. 實作完整性(22.5%)
  4. Demo 展示設計/報告流暢性(22.5%)
  5. 新增：同儕評分(10%)

- PS. 期末專題額外加分
  - 報名校外相關競賽
    - <u>**證明方式**</u>：FTP上傳報名資料及報名成功之Email/報名系統截圖紀錄，須包含：競賽名稱, 作品名稱, 小組成員, 指導老師 (梁, 或可加其他老師))
    - 報名成功：期末專題加分(+2.5分)
    - 獲得名次：再加分(+5~7.5分：佳作/第三名/第二名/第一名)